Universidad de Sevilla  
Escuela Técnica Superior de Ingeniería Informática



**Grado en Ingeniería Informática – Ingeniería del Software**  
**Diseño y Pruebas II**

Curso 2024 – 2025

|  |  |
| --- | --- |
| **Fecha** | **Versión** |
| 3/07/25 | 1.1 |

|  |  |
| --- | --- |
| **Grupo de Prácticas: C1.045** | |
| **Repositorio:** [**https://github.com/DPII-24-25/Acme-ANS-D**](https://github.com/DPII-24-25/Acme-ANS-D) | |
| Lopez Veiga, Juan Carlos | jualopvei@alum.us.es |
|  |  |

Tabla de contenido

[**1.** **Resumen Ejecutivo** 2](#_Toc199162606)

[2. **Tabla de revisiones** 3](#_Toc199162607)

[**3.** **Introducción** 3](#_Toc199162608)

[**4.** **Contenido** 4](#_Toc199162609)

[**5.** **Conclusiones** 16](#_Toc199162610)

[**6.** **Bibliografía** 17](#_Toc199162611)

# **Resumen Ejecutivo**

Este informe de pruebas está dividido en dos capítulos que abordan aspectos fundamentales del proceso de pruebas de nuestro proyecto: pruebas funcionales y pruebas de rendimiento.

El primero detalla los casos de prueba implementados, agrupados según características específicas del sistema. Para cada caso de prueba, se incluye:

- Descripción del Caso de Prueba: Un resumen breve del objetivo del caso de prueba.

- Efectividad en la Detección de Errores: Una evaluación de la capacidad del caso de prueba para detectar errores.

Esto asegura la cobertura de todas las funcionalidades críticas del sistema y documenta la capacidad de cada prueba para identificar fallos, proporcionando una visión clara de la robustez funcional del proyecto.

El segundo se enfoca en las pruebas de rendimiento del sistema, presentando gráficos detallados y análisis estadísticos sobre el tiempo de respuesta al procesar solicitudes en dos computadoras diferentes. Los elementos clave son:

- Gráficos Adecuados: Visualizaciones del desempeño del sistema en términos de tiempos de respuesta.

- Intervalo de Confianza del 95%: Estimaciones precisas y confiables del rendimiento del sistema.

- Contraste de Hipótesis con un 95% de Confianza: Análisis estadístico que compara el rendimiento entre las dos computadoras, determinando cuál ofrece mejor desempeño.

Esto proporciona datos concretos sobre el rendimiento del sistema y ayuda a identificar el entorno de hardware más eficiente, guiando decisiones futuras sobre la infraestructura necesaria.

Estos capítulos ofrecen una evaluación exhaustiva de la calidad y el rendimiento del sistema probado, asegurando la cobertura de todas las áreas críticas y que el proyecto cumple con lo que se espera.

# **Tabla de revisiones**

|  |  |  |
| --- | --- | --- |
| **Fecha** | **Revisión** | **Descripción** |
| 24/05/2025 | 1.0 | Versión Inicial |
| 3/07/25 | 1.1 | Versión Julio |

# **Introducción**

El propósito de este informe es documentar de manera completa los resultados obtenidos a través de las pruebas funcionales y de rendimiento realizadas en nuestro proyecto. Estas pruebas son esenciales en el ciclo de desarrollo de software, ya que garantizan la calidad, fiabilidad y eficiencia del producto final.

En el capítulo dedicado a las pruebas funcionales, se describen en detalle los diferentes casos de prueba implementados, organizados según las características específicas del sistema. Cada caso de prueba incluye una descripción breve y precisa, así como una evaluación de su efectividad en la detección de errores. Este enfoque sistemático asegura que todas las funcionalidades críticas del sistema han sido evaluadas exhaustivamente, proporcionando una visión clara de su robustez y fiabilidad.

Mientras que en el capítulo sobre pruebas de rendimiento se enfoca en el análisis del tiempo de respuesta del sistema al procesar solicitudes, antes y después de implementar los índices. Se presentan gráficos detallados y un intervalo de confianza del 95% para los tiempos de respuesta medidos. Además, se realiza un contraste de hipótesis con un 95% de confianza para determinar cuál de las dos versiones (sin y con índices) evaluadas ofrece un mejor rendimiento. Este análisis no sólo proporciona datos precisos sobre la eficiencia del sistema, sino que también orienta futuras decisiones sobre la infraestructura tecnológica necesaria para optimizar el rendimiento.

# **Contenido**

* 1. **Capítulo de pruebas funcionales**
     1. **Descripción de los casos de prueba .safe**
* *TechnicianMaintenanceRecordListService.java*

- Se accede al listado de los registros de mantenimiento teniendo iniciada una sesión como un technician si se accede a “Mis registros” se verán todos los registros del technician mientras que en el “Todos los registros publicados” podrá ver todos los registros publicados independientemente si son suyos o no.

* *TechnicianMaintenanceRecordShowService*.java

- Se accede a un formulario de un registro de mantenimiento teniendo iniciada una sesión como un technician y solo podrá acceder a las suyas propias o los registros que ya hayan sido publicado por otros technicians.

* *TechnicianMaintenanceRecordCreateService*.java

|  |  |  |
| --- | --- | --- |
| **Atributo** | **Casos** **positivos** | **Casos Negativos** |
| moment | * Sigue el formato Temporal.Type.TIMESTAMP * Entre el 01/01/2000 00:00 y 01/01/2025 00:00 | -Vacío  -Formato incorrecto  -Salir fuera del limite |
| status | * Se elige la opción “PENDING” o “IN\_PROGRESS” | -No se elige opción  -Se elige la opción “COMPLETED” |
| inspectDueDate | * Sigue el formato Temporal.Type.TIMESTAMP * Entre el 01/01/2000 00:00 y 31/12/2200 23:59 * Debe ocurrir después de moment | -Vacío  -Formato incorrecto  -Salir fuera del limite  -Que no exista momento  -Que ocurra antes que moment |
| estCost | * Sigue el formato MONEY * Entre el 0 y 1000000.00 * La moneda debe ser EUR, USD o GBP | -Vacío  -Formato incorrecto  -Salir fuera del limite  -meter una moneda distinta a las listadas |
| moreInfo | * Debe tener como máximo 255 caracteres | -Exceder los 255 caracteres |
| Aircraft | * Seleccionar un Aircraft de los disponibles | -No seleccionar un Aircraft |

* *TechnicianMaintenanceRecordPublishService*.java

|  |  |  |
| --- | --- | --- |
| **Atributo** | **Casos** **positivos** | **Casos Negativos** |
| moment | * Sigue el formato Temporal.Type.TIMESTAMP * Entre el 01/01/2000 00:00 y 01/01/2025 00:00 | -Vacío  -Formato incorrecto  -Salir fuera del limite |
| status | * Se elige la opción “COMPLETED” | -No se elige opción  -No se elige la opción “COMPLETED” |
| inspectDueDate | * Sigue el formato Temporal.Type.TIMESTAMP * Entre el 01/01/2000 00:00 y 31/12/2200 23:59 * Debe ocurrir después de moment | -Vacío  -Formato incorrecto  -Salir fuera del limite  -Que no exista momento  -Que ocurra antes que moment |
| estCost | * Sigue el formato MONEY * Entre el 0 y 1000000.00 * La moneda debe ser EUR, USD o GBP | -Vacío  -Formato incorrecto  -Salir fuera del limite  -meter una moneda distinta a las listadas |
| moreInfo | * Debe tener como máximo 255 caracteres | -Exceder los 255 caracteres |
| Aircraft | * Seleccionar un Aircraft de los disponibles | -No seleccionar un Aircraft |

* Adicionalmente debe tener como mínimo una tarea asociada sino no se podrá publicar el registro.
* *TechnicianMaintenanceRecordUpdateService*.java

|  |  |  |
| --- | --- | --- |
| **Atributo** | **Casos** **positivos** | **Casos Negativos** |
| moment | * Sigue el formato Temporal.Type.TIMESTAMP * Entre el 01/01/2000 00:00 y 01/01/2025 00:00 | -Vacío  -Formato incorrecto  -Salir fuera del limite |
| status | * Se elige la opción “COMPLETED”, “PENDING” o “IN\_PROGRESS” | -No se elige opción  -No se elige la opción “COMPLETED” |
| inspectDueDate | * Sigue el formato Temporal.Type.TIMESTAMP * Entre el 01/01/2000 00:00 y 31/12/2200 23:59 * Debe ocurrir después de moment | -Vacío  -Formato incorrecto  -Salir fuera del limite  -Que no exista momento  -Que ocurra antes que moment |
| estCost | * Sigue el formato MONEY * Entre el 0 y 1000000.00 * La moneda debe ser EUR, USD o GBP | -Vacío  -Formato incorrecto  -Salir fuera del limite  -meter una moneda distinta a las listadas |
| moreInfo | * Debe tener como máximo 255 caracteres | -Exceder los 255 caracteres |
| Aircraft | * Seleccionar un Aircraft de los disponibles | -No seleccionar un Aircraft |

* *TechnicianMaintenanceRecordDeleteService*.java

-Se borran los registros de mantenimiento teniendo iniciado sesión de su correspondiente technician siempre y cuando no tenga alguna asociación de tareas.

//////////-------------------------------///////////////////

* *TechnicianTaskAssociationListService.java*

- Se accede al listado de las asociaciones de tareas teniendo iniciada una sesión como un technician y entrando en su registro correspondiente.

* *TechnicianTaskAssociationShowService*.java

- Se accede a un formulario de una asociación teniendo iniciada una sesión como un technician y en su registro correspondiente.

* *TechnicianTaskAssociationCreateService*.java

|  |  |  |
| --- | --- | --- |
| **Atributo** | **Casos** **positivos** | **Casos Negativos** |
| task | * Eligir un Task de las diponibles | -Elegir Ninguno |

* *TechnicianTaskAssociationDeleteService*.java

-Se borran las asociaciones de tareas teniendo iniciado sesión de su correspondiente technician y entrando en su registro correspondiente.

//////////-------------------------------///////////////////

* *TechnicianTaskListService.java*

- Se accede al listado de las tareas teniendo iniciada una sesión como un technician si se accede a “Mis tareas” se verán todas las tareas del technician mientras que en el “Todas las tareas publicadas” podrá ver todas las tareas publicadas independientemente si son suyos o no.

* *TechnicianTaskShowService*.java

- Se accede a un formulario de una tarea teniendo iniciada una sesión como un technician y solo podrá acceder a las suyas propias o las tareas que ya hayan sido publicado por otros technicians.

* *TechnicianTaskCreateService*.java & *TechnicianTaskPublishService*.java & *TechnicianTaskUpdateService*.java

|  |  |  |
| --- | --- | --- |
| **Atributo** | **Casos** **positivos** | **Casos Negativos** |
| type | * Se elige la opción “MAINTENANCE”, “REPAIR”, “SYSTEM\_CHECK” o “INSPECTION” | -No se elige opción |
| description | * Debe tener como máximo 255 caracteres | -Vacio/En blanco  -Exceder los 255 caracteres |
| deadline | * Sigue el formato Temporal.Type.TIMESTAMP * Entre el 01/01/2000 00:00 y 31/12/2200 23:59 | -Vacío  -Formato incorrecto  -Salir fuera del limite |
| priority | * Integer que debe valer entre 0 y 10 | -Vacío  -Formato incorrecto  -Salir fuera del limite |
| estimatedDuration | * Integer que debe valer entre 0 y 1000 | -Vacío  -Formato incorrecto  -Salir fuera del limite |

* *TechnicianTaskDeleteService*.java

-Se borran las tareas teniendo iniciado sesión de su correspondiente technician.

* + 1. **Descripción de los casos de prueba .hack**
* *TechnicianMaintenanceRecordListService.java*

- **Wrong realm**: se intenta acceder al listado de registros de mantenimiento como un usuario sin identificar y otro rol (No technician) cualquiera.

* *TechnicianMaintenanceRecordShowService.java*

- **Wrong realm**: se intenta acceder al formulario de un registro de mantenimiento como un usuario sin identificar y otro rol (No technician) cualquiera.

- **Wrong user**: se intenta acceder al formulario de un registro de mantenimiento (no publicado) con un technician diferente al propietario de este.

* *TechnicianMaintenanceRecordCreateService.java*

- **Wrong realm**: se intenta crear un registro de mantenimiento como un usuario sin identificar y otro rol (No technician) cualquiera.

- **Navigation attribute**: se introduce un valor ilegal a un atributo de navegación y luego se intenta entregar el formulario.

* *TechnicianMaintenanceRecordUpdateService.java*

- **Wrong realm**: se intenta crear un registro de mantenimiento como un usuario sin identificar y otro rol (No technician) cualquiera.

- **Wrong user**: se intenta actualizar un registro de mantenimiento en estado de borrador con un technician diferente al propietario de este.

- **Wrong action**: se intenta actualizar un registro de mantenimiento en estado publicado con el technician propietario de este.

- **Navigation attribute**: se introduce un valor ilegal a un atributo de navegación y luego se intenta entregar el formulario.

* *TechnicianMaintenanceRecordPublishService.java*

- **Wrong realm**: se intenta crear un registro de mantenimiento como un usuario sin identificar y otro rol (No technician) cualquiera.

- **Wrong user**: se intenta actualizar un registro de mantenimiento en estado de borrador con un technician diferente al propietario de este.

- **Wrong action**: se intenta volver a publicar un registro de mantenimiento en estado publicado con el technician propietario de este.

- **Navigation attribute**: se introduce un valor ilegal a un atributo de navegación y luego se intenta entregar el formulario.

* *TechnicianMaintenanceRecordDeleteService.java*

- **Wrong realm**: se intenta crear un registro de mantenimiento como un usuario sin identificar y otro rol (No technician) cualquiera.

- **Wrong user**: se intenta actualizar un registro de mantenimiento en estado de borrador con un technician diferente al propietario de este.

- **Wrong action**: se intenta eliminar un registro de mantenimiento en estado publicado con el technician propietario de este.

//////////-------------------------------///////////////////

* *TechnicianTaskAssociationListService.java*

- **Wrong realm**: se intenta acceder al listado de asociaciones de tareas como un usuario sin identificar y otro rol (No technician) cualquiera.

- **Wrong user**: se intenta actualizar una asociación de tareas de un registro en estado de borrador con un technician diferente al propietario de este.

* *TechnicianTaskAssociationShowService.java*

- **Wrong realm**: se intenta acceder al formulario de una asociación de tareas como un usuario sin identificar y otro rol (No technician) cualquiera.

- **Wrong user**: se intenta acceder al formulario de una asociación de tareas (no publicado) con un technician diferente al propietario de este.

* *TechnicianTaskAssociationCreateService.java*

- **Wrong realm**: se intenta crear una asociación de tareas como un usuario sin identificar y otro rol (No technician) cualquiera.

- **Wrong user**: se intenta crear una asociación de tareas en estado de un registro en borrador con un technician diferente al propietario del registro.

- **Wrong action**: se intenta crear una asociación de tareas de un registro en estado publicado con el technician propietario de este.

- **Navigation attribute**: se introduce un valor ilegal a un atributo de navegación y luego se intenta entregar el formulario.

* *TechnicianTaskAssociationDeleteService.java*

- **Wrong realm**: se intenta crear una asociación de tareas como un usuario sin identificar y otro rol (No technician) cualquiera.

- **Wrong user**: se intenta actualizar una asociación de tareas en estado de un registro en borrador con un technician diferente al propietario de este.

- **Wrong action**: se intenta eliminar una asociación de tareas de un registro en estado publicado con el technician propietario de este.

//////////-------------------------------///////////////////

* *TechnicianTaskListService.java*

- **Wrong realm**: se intenta acceder al listado de tareas como un usuario sin identificar y otro rol (No technician) cualquiera.

* *TechnicianTaskShowService.java*

- **Wrong realm**: se intenta acceder al formulario de una tarea como un usuario sin identificar y otro rol (No technician) cualquiera.

- **Wrong user**: se intenta acceder al formulario de una tarea (no publicado) con un technician diferente al propietario de este.

* *TechnicianTaskCreateService.java*

- **Wrong realm**: se intenta crear una tarea como un usuario sin identificar y otro rol (No technician) cualquiera.

- **Navigation attribute**: se introduce un valor ilegal a un atributo de navegación y luego se intenta entregar el formulario.

-**Read-only**: se trata de modificar los atributos read only de un registro ya publicado.

* *TechnicianTaskUpdateService.java*

- **Wrong realm**: se intenta crear una tarea como un usuario sin identificar y otro rol (No technician) cualquiera.

- **Wrong user**: se intenta actualizar una tarea en estado de borrador con un technician diferente al propietario de este.

- **Wrong action**: se intenta actualizar una tarea en estado publicado con el technician propietario de este.

-**Read-only**: se trata de modificar los atributos read only de un registro ya publicado.

- **Navigation attribute**: se introduce un valor ilegal a un atributo de navegación y luego se intenta entregar el formulario.

* *TechnicianTaskPublishService.java*

- **Wrong realm**: se intenta crear una tarea como un usuario sin identificar y otro rol (No technician) cualquiera.

- **Wrong user**: se intenta actualizar una tarea en estado de borrador con un technician diferente al propietario de este.

- **Wrong action**: se intenta volver a publicar una tarea en estado publicado con el technician propietario de este.

- **Navigation attribute**: se introduce un valor ilegal a un atributo de navegación y luego se intenta entregar el formulario.

-**Read-only**: se trata de modificar los atributos read only de un registro ya publicado.

* *TechnicianTaskDeleteService.java*

- **Wrong realm**: se intenta crear una tarea como un usuario sin identificar y otro rol (No technician) cualquiera.

- **Wrong user**: se intenta actualizar una tarea en estado de borrador con un technician diferente al propietario de este.

- **Wrong action**: se intenta eliminar una tarea en estado publicado con el technician propietario de este.

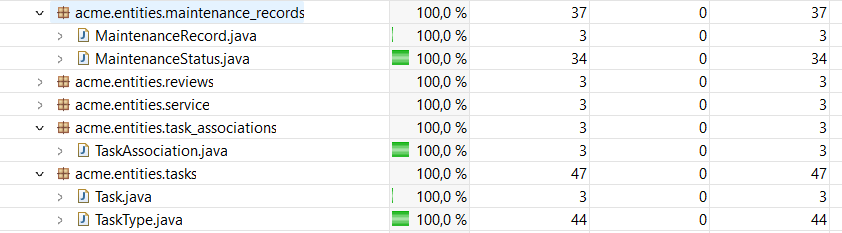
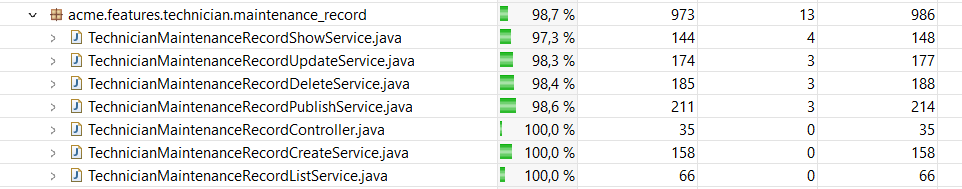
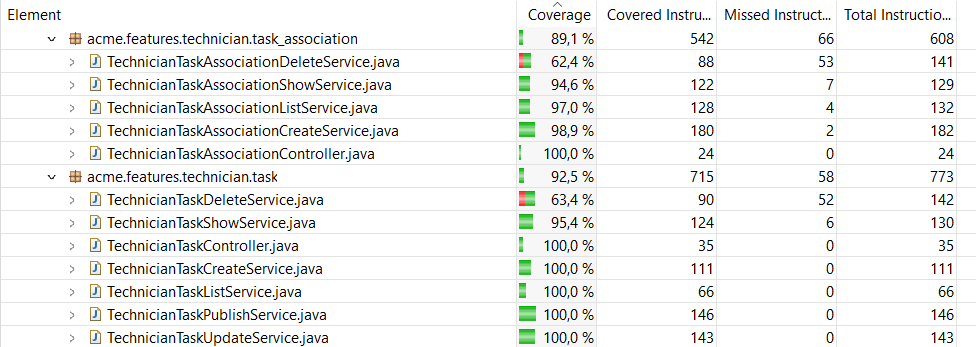
* + 1. **Bugs encontrados**

Tras la realización de todos los tests anteriormente mencionados se encontraron los siguientes bugs:

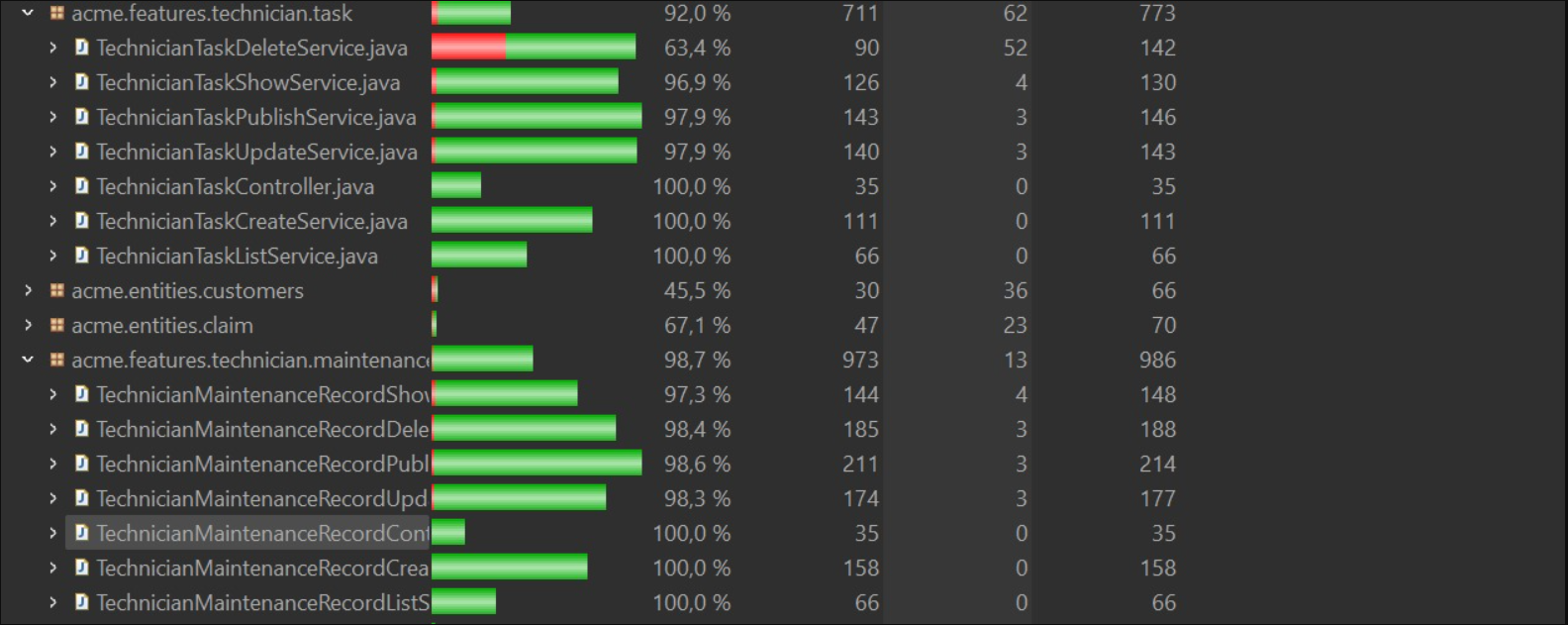
|  |  |
| --- | --- |
| **BUGS** | **SOLUCION ENCONTRADA** |
| Al tratar borrar un “MaintenanceRecord” con algún “TaskAssociation” devolvía panic. | Implementar una validación en TechnicianMaintenanceRecordDeleteService en donde no te permite borrar un Records si existe algún TaskAssociation. |
| Era posible asociar una tarea a un “MaintenanceRecord” que no pertenece al technician mediante POST hacking. | Mejorar el authorize() del TechnicianTaskAssociationCreateService para que no pasase. |

* + 1. **Cobertura**

Cobertura en mi PC:

****

Cobertura en otro PC

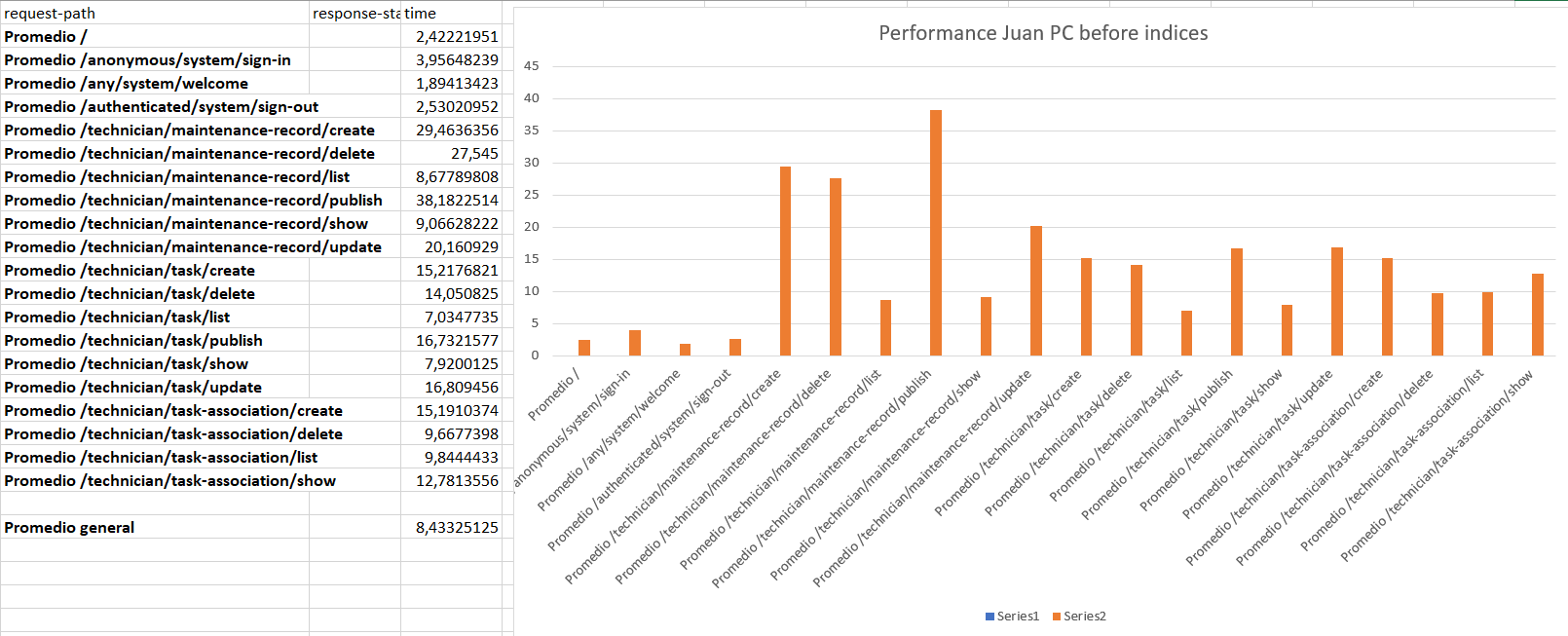
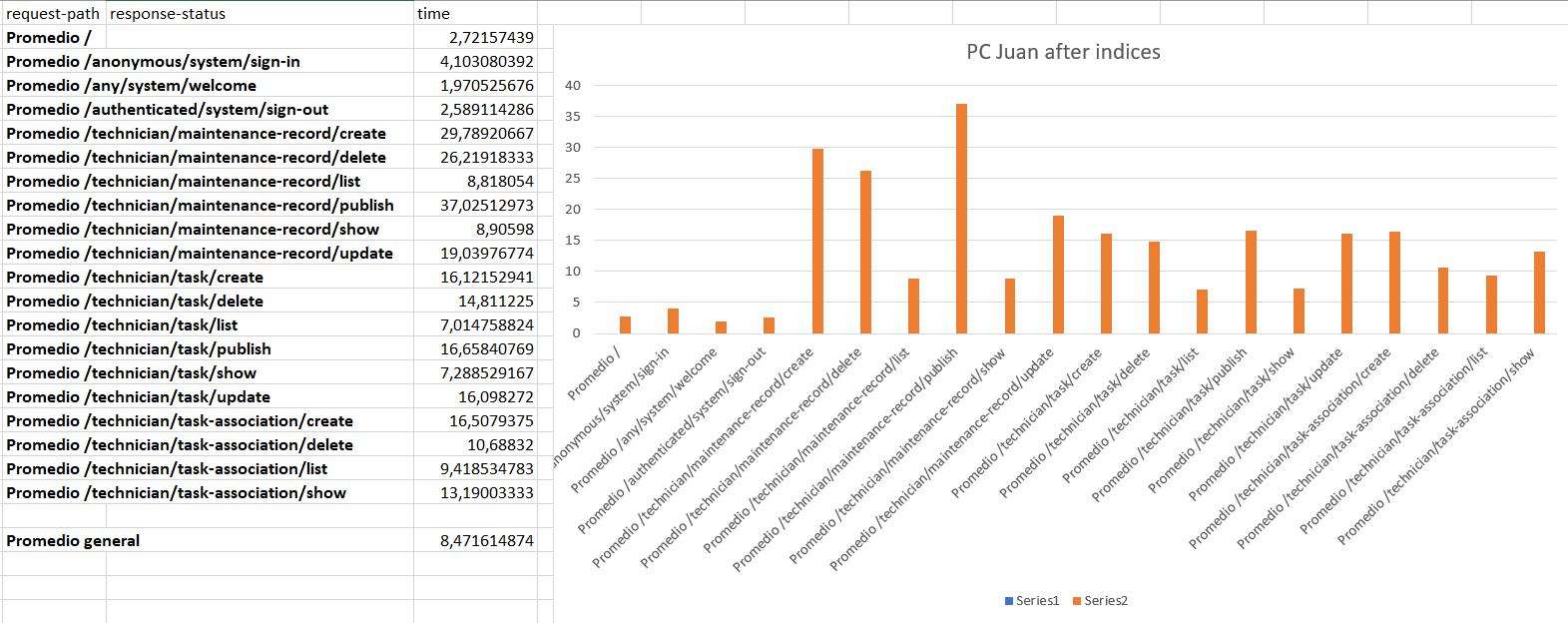


Haciendo un análisis de la cobertura se puede observar que lo que hace disminuir la cobertura es principalmente el método “unbind” de la features

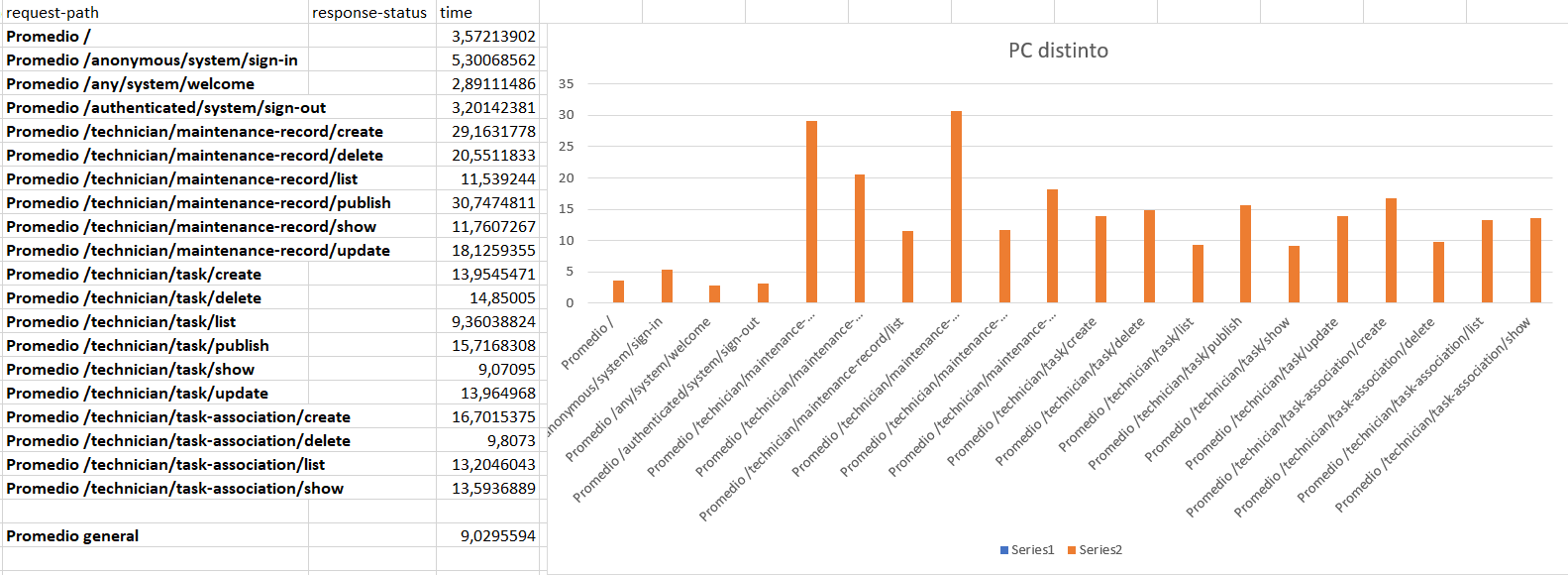
TechnicianTaskDeleteService.java y TechnicianTaskAssociation DeleteService.java

* 1. **Capítulo de pruebas de rendimiento**

**4.2.1 Análisis promedios y gráficas**

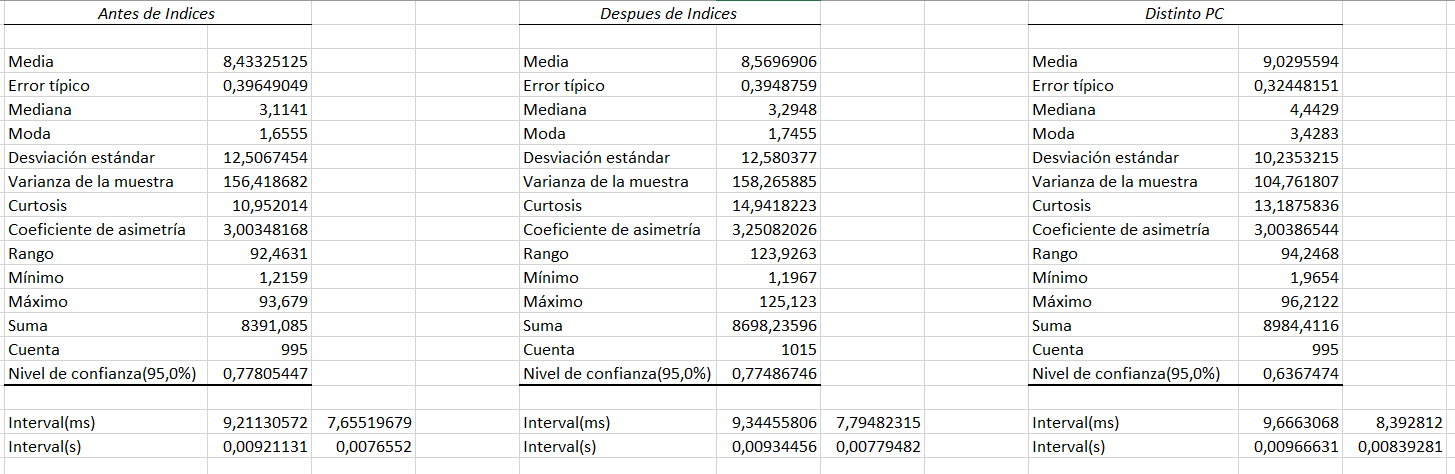
**** ****

El performance antes y despues de añidar los indices es nominal, se podria atribuir su ligero aumento a la variabilidad aleatoria de los datos de prueba.

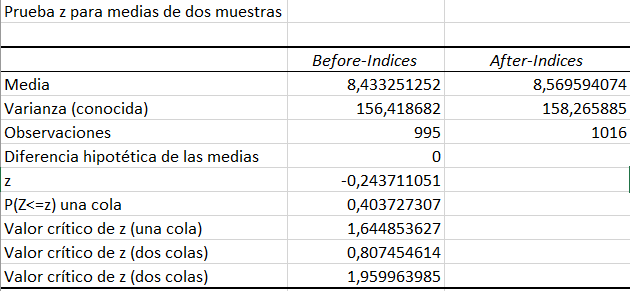
****

Probando en un distinto se ce que ha aumentado el promedio general.

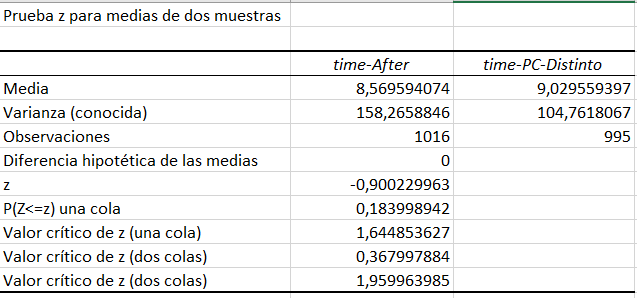
**4.2.2 Análisis estadística descriptiva**

****

**4.2.3 Análisis Z-test**

****

Como podemos ver por el valor de “Valor crítico de z(dos colas) = 0.807…” los cambias no han aportado ningún cambio significativo a la hora de procesar los tests.



Como podemos ver por el valor de “Valor crítico de z(dos colas) = 0.367…” los cambias han aportado un cambio significativo a la hora de procesar los tests ya que ese valor se encuentra por debajo de α=0.5 ahora debemos comparar los promedios y como el de After-indices tiene un promedio mas bajo que el de PC-distinto indica que ha empeorado el tiempo de ejecución al probarlo en el ordenador distinto.

# **Conclusiones**

El examen minucioso de las pruebas funcionales y de rendimiento llevadas a cabo en nuestro proyecto muestra resultados favorables en cuanto a la eficiencia y la fiabilidad del sistema. En lo que respecta a las pruebas funcionales, se implementaron y documentaron exhaustivamente los casos de prueba, organizados según las características específicas del sistema.

En relación con las pruebas de rendimiento, se llevaron a cabo mediciones detalladas del tiempo de respuesta del sistema al procesar solicitudes. Los resultados incluyen gráficos y análisis estadísticos, junto con un intervalo de confianza del 95% para los tiempos de respuesta. La prueba de hipótesis realizada con un 95% de confianza permitió concluir que las acciones efectuadas no han mejorado significativamente el sistema, y que la peoría observada podría ser resultado de la variabilidad aleatoria de los datos de prueba.

# **Bibliografía**

Intencionalmente en blanco.